MDSML: An XML Binding to the Grid Object Specification

A proposal to be discussed as part of the Gridforum

Version: 0.3.0

Gregor von Laszewski and Peter Lane

Mathematics and Computer Science Division at Argonne National Laboratory
9700 S. Cass Avenue
Argonne, IL 60439 U.S.A.

DRAFT: This is not a final draft; please send comments to gregor@mcs.anl.gov
Editor: Gregor von Laszewski

Contents

1 Introduction

The Metacomputing Directory Services Markup Language (MDSML) allows one to describe objects specified with the Grid Object Specification (GOS) [3]. The purpose of GOS is to enable to define a large number of objects that are used as part of the Grid Information Services. Since objects are typically defined by various groups, it is essential that objects be defined constantly and easily. GOS provides one solution. With the advent of XML [1] and sophisticated tools [4] that can be used to parse and translate XML documents, it is of great advantage to provide an XML binding to GOS. Besides the definition of schemas through GOS, MDSML allows one also to formulate entries that follow the specifications defined by the language.

We have developed prototype tools that can convert MDSML into various output formats, including HTML, DSML, ASCII, and LaTeX. We expect to extend these prototypes to include tools that also can verify entries. Because of the need for scalable Grid Information Services the ability of client-side, verification of entries is of utmost importance to reduce the load on information servers.

1.1 Notation

The terminology of [3] is used in this documentation. We assume the reader is sufficiently familiar with this notation.

1.2 Relationship to DSML

Originally MDSML was called gosML. Recently we changed the name to reflect the close relationship to the Directory Services Markup Language (DSML) [2], defined as industry standard. MDSML was defined before the first official DSML specification was released, but we have made some changes to the original MDSML to bring the languages closer together.

One of the most challanging tasks an information services group faces is to specify the objects and their attributes so that uniform infrastructures in a heterogeneous environment can be created. We feel that the current DSML language does not allow one to define all the constructs that are necessary to define, maintain, and extend schemas between a large heterogeneous group of people defining and maintaining a distributed information service. To underline this, we quote from [2] :

> It is not an initial goal of DSML to specify the attributes that all directories must contain, or the method with which the directory information is accessed from the directory. The expectation is that standard protocols (such as LDAP), proprietary access protocols (such as Novell's NDAP) and proprietary APIs (such as Microsoft's ADSI) could produce DSML documents as an optional output.

Following this intention of the use of DSML, one can produce a translator that transforms MDSML documents to DSML. Although the inverse is also possible, it will not allow one to formulate all the semantic information that is defined in MDSML through a DSML document.

2 Example

An MDSML document can have three different objects: object classes, aggregate classes, and entries. Object classes and aggregate classes are used to define the structure in which entries are defined. We follow the example as used in [3].

2.1 Aggregate Class

It is straightforward to transform an aggregate class specified in the GOS format to MDSML. The example below demonstrates how to define an aggregate object.

```
<mdsml:aggregateclass name=''Location'' namespace=''Grid''>
  <mdsml:description>
  This aggregate object defines a geographical location. A
  geographical location is often used for the graphical display
  of prganizations or compute resources on a map.
  </mdsml:description>
  <mdsml:attribute name=''static''
             type=''cisboolean''
             occurence=''single''
             required=''true''>
        <mdsml:description>
        In case the location is fixed, static is set to true
        otherwise false.
        </description>
  </mdsml:attribute>
  <mdsml:attribute name=''latitude''
             type=''cisfloat''
             occurence=''single''
             required=''false''>
        <mdsml:description>
        Specifies the value of the latitude in the form
        ''degree minute second''
        </mdsml:description>
  </mdsml:attribute>
  <mdsml:attribute name=''longitude''
             type=''cisfloat''
             occurence=''single''
             required=''false''>
        <mdsml:description>
        Specifies the value of the longitude in the form
        ''degree minute second''
        </mdsml:description>
   </mdsml:attribute>
</mdsml:aggregateclass>
```

2.2 Object Class

An object class is as simple to define as an aggregate class. In contrast to an aggregate class, however, the object class contains additional tags, such as *superior, rdn, childof*, and *aggregates*. We have chosen the name *superior* to represent the SUBCLASS OF block in GOS. The choice of name is motivated by the fact that the tag is called in such a manner in DSML.

In the following example we have left out further descriptions. It is obvious that a correct usage of the definition of an object contains *detailed* descriptive tags to describe the usage of an attribute or the object itself.

```
<mdsml:objectclass name=''ComputeResource'' namespace=''Grid''>

  <mdsml:description>
        ...
  </mdsml:description>

  <mdsml:superior  name=''PhysicalResource'' namespace=''Grid''/>
  <mdsml:rdn name=''hn'' expansion=''hostName''/>

  <mdsml:childof>
     <mdsml:objectname name=''organizationalUnit'' namespace=''Grid''>
     <mdsml:objectname name=''organization'' namespace=''Grid''>
  </mdsml:childof>

  <mdsml:aggregates>
     <mdsml:objectname name=''location'' namespace=''Grid''>
  </mdsml:aggregates>

  <mdsml:attribute name=''canonicalSystemName''
             type=''cis''
             occurence=''single''
             required=''true''>
        <mdsml:description>
        ...
        </mdsml:description>
   </mdsml:attribute>

 <mdsml:attribute name=''manufacturer''
             type=''cis''
             occurence=''single''
             required=''true''>
        <mdsml:description>
        ...
        </mdsml:description>
   </mdsml:attribute>

 <mdsml:attribute name=''manufacturer''
             type=''cis''
             occurence=''single''
             required=''true''>
        <mdsml:description>
        ...
        </mdsml:description>
   </mdsml:attribute>

 <mdsml:attribute name=''model''
             type=''ces''
             occurence=''single''
```

```
            required=''true''>
        <mdsml:description>
        ...
        </mdsml:description>
   </mdsml:attribute>

 <mdsml:attribute name=''diskDrive''
            type=''cis''
            occurence=''multiple''
            required=''false''>
        <mdsml:description>
        ...
        </mdsml:description>
   </mdsml:attribute>

</mdsml:objectclass>
```

2.3 Entry

Object classes and aggregates together describe the schema of the information to be represented. An instantiation of data that is described via a schema is often called an entry. An entry is defined by its distinguished name, and attributes that can have multiple values depending on its schema definition. Multiple entries can be included in a document that is defined by MDSML. A client library must be able to verify whether entries follow the definition of a particular schema.

```
<mdsml:entry dn=''hn=computer.anl.gov,
                 o=Argonne National Laboratory,
                 o=Globus, C=US''>
    <mdsml:entry-oc>
      <mdsml:value> Grid::ComputeResource </mdsml:value>
    </mdsml:entry-oc>

    <mdsml:entry-at name=''canonicalSystemName''>
      <mdsml:value> linux-386-2.012   <mdsml:value>
    </mdsml:entry-at>
    <mdsml:entry-at name=''manufacturer''>
      <mdsml:value> IBM <mdsml:value>
    </mdsml:entry-at>
    <mdsml:entry-at name=''model''>
      <mdsml:value> IBM <mdsml:value>
    </mdsml:entry-at>
    <mdsml:entry-at name=''serialNumber''>
      <mdsml:value> T300Th67617653276 <mdsml:value>
    </mdsml:entry-at>
    <mdsml:entry-at name=''static''>
      <mdsml:value> false <mdsml:value>
    </mdsml:entry-at>
    <mdsml:entry-at name=''longitude''>
      <mdsml:value> 55 12 3 <mdsml:value>
```

```
    </mdsml:entry-at>
    </mdsml:entry-at name=''latitude''>
      <mdsml:value> 30 4 5 <mdsml:value>
    </mdsml:entry-at>
</mdsml:entry>
```

3 MDSML Document Type Definition

This section contains the full document type definition (DTD) of MDSML.

```
<!-- VERSION October, 2000 -->

<!ELEMENT mdsml (objectclass | aggregateclass | entry)*>

<!ELEMENT objectclass ( (description)?,
(superior)?,
                        (rdn),
                        (childof)?,
                        (aggregates)?,
(attribute)*)>
<!ATTLIST objectclass
  name CDATA #REQUIRED
  namespace CDATA "Grid"
  oid CDATA #IMPLIED
>

<!ELEMENT aggregateclass ( (description)?,
                        (childof)?,
                        (aggregates)?,
(attribute)*)>
<!ATTLIST aggregateclass
  name CDATA #REQUIRED
  namespace CDATA "Grid"
  oid CDATA #IMPLIED
>
<!ELEMENT description (#PCDATA)>

<!ELEMENT oid EMPTY>
<!ATTLIST oid
  value CDATA #REQUIRED
>
<!ELEMENT superior EMPTY>
<!ATTLIST superior
  name CDATA #REQUIRED
  namespace CDATA "Grid"
>
<!ELEMENT rdn EMPTY>
```

```
<!ATTLIST rdn
  name CDATA #REQUIRED
  expansion CDATA #REQUIRED
>
<!ELEMENT childof (objectname)+>

<!ELEMENT aggregates (objectname)+>

<!ELEMENT objectname EMPTY>
<!ATTLIST objectname
  name CDATA #REQUIRED
  namespace CDATA "Grid"
>
<!ELEMENT attribute (description)>
<!ENTITY  % STRTYPE "(cis | ces | bin | int | dn
    | cisfloat | cisdate | cisboolean | tel | url)">
<!ENTITY  % BOOLEAN "(true | false)">
<!ENTITY  % OCCUROP "(single | multiple)">
<!ATTLIST attribute
  name CDATA #REQUIRED
  type %STRTYPE; "cis"
  oid CDATA #IMPLIED
  required %BOOLEAN; "false"
  occurrence %OCCUROP; "single"
>
<!ELEMENT entry  ( (entry-oc),
  (entry-at)* )>
<!ATTLIST entry
  dn CDATA #REQUIRED
>
<!ELEMENT entry-oc (value+)>
<!ELEMENT entry-at (value+)>
<!ATTLIST entry-at
  name CDATA #REQUIRED
>
<!ELEMENT value (#PCDATA)* >
```

4 Conversion Tools

Included here is an ascii version of the manpage for the mdsml-converter tool which we have developed.

```
mdsml-converter(1) User Manuals mdsml-converter(1)
NAME
 mdsml-converter <-> Convert MDSML documents to other formats
SYNOPSIS
 mdsml-converter --input input-file [options]
```

```
DESCRTPTION
 mdsml-converter is a bourne shell script wrapper for
 org.globus.mdsml.Converter which comes packaged with the
 Java CoG Kit. Converter is a Java (SDK2) application that
 reads an MDSML (XML) document, converts the information to
 another format, and then writes the converted document to
 either standard out or a user-specified file. Because
 this is a Java program, the jar file containing needed
 classes must be located. To do this, mdsml-converter
 requires that the COG_INSTALL_PATH environment variable be
 set to the installation directory of the Java CoG Kit.
 Currently, the supported output formats include DSML, GOS,
 MDS Attributes, MDS Objectclasses, an HTML version of GOS,
 a LaTex version of GOS, LDAPv3 subschema, as well as DIT,
 inheritance, and aggregate graphs in dot input format.
OPTIONS
 --format output-format
            The output format (gos | attribute | objectclass |
            html | latex | dsml | dit | inheritance | ldapv3 |
            aggregate)
 --help
            Display the usage screen.
 --output output
            The output file for the converted document (default
            is stdout).
 --seperator seperator
            The namespace/type seperator (default is '::').
 --oid-db oid-database
            Specify the ldap "at" file which serves as a
            database for retrieving OIDs. This option also
            flags the converters to write out OIDs. If this
            option is not used, OIDs will not be displayed.
 --with-namespace|--without-namespace
            Do or do not output namespaces for formats that may
            contain them (default is without).
 --with-alternate|--without-alternate
            Do or do not generate an alternate output (default
            is without). This is currently only used in LaTex.
OBTAINING THE SOFTWARE
 mdsml-converter is distributed with the Java CoG Kit
 (CoG). To obtain CoG, visit the CoG website at
 http://www.globus.org/cog.
BUILDING AND INSTALLING THE SOFTWARE
 To build the Java CoG Kit, a traditional configure...
 make...make install sequence is employed. If you only
 wish to build the packages needed to run mdsml-converter,
 perform the following configure command substituting an
 appropritate installation prefix:
```

```
 configure --prefix=<installation prefix> --without-applet
 --without-common --without-example --without-gara
 --without-gram --without-io --without-mds --without-myproxy
 --without-rsl --without-security --without-tools
 Perform a make to build the selected packages.
 Perform a make install to jar up all the compiled classes
 and install the software under the given installation
 prefix directory.  Finally, you must make sure the
 COG_INSTALL_PATH environment variable is set appropriately
 to the installation prefix directory. This will allow the
 script to locate the appropriate java classes.
SAMPLE EXECUTION
 mdsml-converter --input input.mdsml --output output.dsml
 --format dsml
MDSML DTD
 In order to validate and parse an MDSML document, the
 document must point to a valid MDSML DTD in it's DOCTYPE
 declaration. The latest DTD can be found in the CoG
 distribution's "include" directory or on the Gridforum
 GIS Working Group's web site (http://www.mcs.anl.gov/gridforum/gis).
KNOWN BUGS AND LIMITATIONS
TO DO (for developers)
AUTHORS
 The main core of the development was done by Peter Lane
 <lane@mcs.anl.gov>. HTML enhancements and LaTex help was
 provided by Gregor von Laszewski <gregor@mcs.anl.gov>.
```

A Acknowledgement

An earlier but incomplete prototype of gosML was defined by X. Peng and Gregor von Laszewski.

References

[1] GRAH, I. S., AND QUIN, L. *XML Specificaton Guide*. Wiley, 1999.

[2] TAUBER, J., HAY, T., BEAUVAIS, T., BURATI, M., AND ROBERTS, A. Directory Services Markup Language (DSML). http://www.dsml.org, 2 Dec. 1999.

[3] VON LASZEWSKI, G., FITZGERALD, S., DIDIER, B., SCHUCHARDT, K., AND LANE, P. *GOS: Defining Schemas for the Grid Information Services.* Gridforum, http://www-unix.mcs.anl.gov/gridforum/gis, February 2000. GIS-WG-1.

[4] xml.com. http://www.xml.com/xml/pub.